**UNIVERSIDAD AUTÓNOMA DE MADRID**

**ESCUELA POLITÉCNICA SUPERIOR**


**Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación**

# TRABAJO FIN DE GRADO

## DESARROLLO DE MEJORAS EN SISTEMA DE MONITORIZACIÓN VoIP

**JULIO 2016**


**Rubén Gómez Moreno**
**Tutor: David Muelas Recuenco**
**Ponente: Jorge Enrique López de Vergara Méndez**

# DESARROLLO DE MEJORAS EN SISTEMA DE MONITORIZACIÓN VoIP

**AUTOR: Rubén Gómez Moreno**
**TUTOR: David Muelas Recuenco**
**PONENTE: Jorge Enrique López de Vergara Méndez**

**Laboratorio HCPN**
**Dpto. de Tecnología Electrónica y de las Comunicaciones**
**Escuela Politécnica Superior**
**Universidad Autónoma de Madrid**
**Julio de 2016**

# Resumen (castellano)

En las últimas décadas, las posibilidades de convergencia y abaratamiento de servicios han hecho que numerosos Proveedores de Servicios apuesten por el uso de las tecnologías de Voz sobre IP como alternativa a la telefonía tradicional. No obstante, la introducción de estos servicios multimedia sobre redes IP entraña una serie de problemas relativos a la prestación de ciertos niveles de calidad de servicio a los usuarios, ocasionados en gran medida por las políticas *best effort* de conmutación de paquetes que siguen este tipo de redes. Esta situación compromete la calidad de experiencia de los usuarios finales, que es un punto de importancia capital para los operadores.

Con el fin de gestionar correctamente los servicios de Voz sobre IP, se han desarrollado diversas herramientas de análisis de red que permiten controlar y monitorizar las principales métricas de calidad. En este Trabajo Final de Grado, se abordan la realización de una serie de mejoras sobre el sistema de análisis, VoIPCallMon [1], desarrollado por el grupo de Redes y Computación de Altas prestaciones (HPCN) de la Universidad Autónoma de Madrid. Específicamente, se presentan la implementación de nuevas métricas de calidad (Post Dial Delay), una mejora de la estimación de las métricas de calidad de experiencia que ya proporciona la herramienta, y una evaluación de librerías de transcripción de audio que pueden permitir que el sistema se complemente con la opción de almacenar el contenido de las conversaciones en formato de texto. Estas mejoras se integran en el sistema VoIPCallMon con el fin de (i) aumentar el rango de alarmas detectadas vinculadas a las nuevas métricas de calidad de servicio; (ii) mejorar la efectividad de la herramienta al adecuar las estimaciones de las métricas de calidad de experiencia al funcionamiento real de las transmisiciones de voz en Internet; y (iii) ampliar el rango de aplicabilidad del sistema, que podrá aportar una visión integral de la calidad de servicio prestado con la inclusión de módulos para transcripción de audio en ámbitos como, por ejemplo, *call centers* empresariales.

Para presentar el desarrollo de estas mejoras, se proporciona una descripción de los principales elementos que aparecen en los despliegues típicos de Voz sobre IP, y la estructura y modo de funcionamiento del sistema VoIPCallMon. Esta primera parte del trabajo motiva los requisitos y decisiones de diseño determinados para las mejoras desarrolladas sobre el sistema. Finalmente, se muestran los resultados de evaluación obtenidos describiendo las características del proceso de pruebas para garantizar la robustez de los resultados, y se resaltan las principales lecciones aprendidas para el futuro desarrollo de este tipo de herramientas.

# Palabras clave (castellano)

Voz sobre IP (VoIP), VoIPCallMon, *Post Dial Delay* (PDD), *Mean Opinion Score* (MOS), *Session Initiation Protocol* (SIP), *Real-Time Transport Protocol* (RTP).

# Abstract (English)

In the last decades, the possibilities for convergence and cheapening of services have caused that most Service Providers bet on using Voice over IP technologies as an alternative to the traditional telephony.

However, the introduction of these multimedia services over IP involves several problems regarding the quality levels of the services which are offer to end users. Particularly, the ¨best effort¨ packet switching policies in this type of networks causes most of the problems that arise during the migration to Voice over IP technologies. This situation jeopardizes the quality of experience for users, which is one of the most important matters for operators.

In this light, many network analysis tools have been developed to properly manage Voice over IP services bycontrolling and monitoring the main quality metrics that are used to evaluate both the quality of service and experience. In this Final Project, we will present the development of severalimprovements on one particular analysis system, VoIPCallMon, developed by theHigh-Performance Computing and Networking research group (HPCN) at the Universidad Autónoma de Madrid.

Specifically, we will explain the implementation of new quality metrics (namely, the Post Dial Delay), the improvement of the estimated quality of experience metrics that the tool provides, and the results of our evaluation of audio transcription libraries that can expand the functionality of the system by allowing the retention of conversations in text files.

These improvements have been integrated in the VoIPCallMon system to (i) increase the range of detected alarms related to new quality metrics of service; (ii) improve the effectiveness of this tool by adapting the estimated quality of experience metrics to the real performance of internet voice transmission; and (iii) extend the range of applicability of the system in areas such as, for instance, corporate call centers, with the inclusion of modules for audio transcription.

In order to present the results of our work, we will provide a description of the main elements that appear in typical Voice over IP deployments, and the structure and way of operation of VoIPCallMon. This first part leads to the definition of the requirements and design decisions for the improvements that will be developed on the system. Finally, wewill show the obtained results with comprehensive descriptions of the characteristics of the testing process to guarantee the strength of the results and highlighting the main lessons learned for the future development of these type of tools.

# Keywords (inglés)

Voz over IP (VoIP), VoIPCallMon, *Post Dial Delay* (PDD), *Mean Opinion Score* (MOS), *Session Initiation Protocol* (SIP), *Real-Time Transport Protocol* (RTP).

# *Agradecimientos*

Durante este TFG, he disfrutado aprendiendo, y he vivido una bonita experiencia. No todo ha sido bueno por supuesto, y también he tenido momentos malos, de nervios, de desesperación, de agobio, etc, pero de los cuales al final he salido exitoso, y la mayor parte de las veces gracias a mi tutor, David.
Es por ello, que antes que a nadie, quiero agradecerle su gran labor a lo largo de todos estos meses con su apoyo constante, su rápida respuesta siempre en cualquier día y momento, su infinita paciencia conmigo ayudándome con cada bache, con cada duda o problema que he tenido, por pequeño que fuese, y por todo lo que he aprendido de él en este trabajo. Muchas gracias por todo.

También me gustaría agradecerte Jorge, el haberme brindado la oportunidad de trabajar en este TFG; ya que me ha gustado mucho y he aprendido mucho más sobre este campo que tanto me gustó ya en las asignaturas de Redes, el de la voz sobre IP.

Quiero agradecer también a mi padre, y a mis amigas Sara y Fany, su apoyo a lo largo no sólo de este TFG, sino de toda la carrera, ya que me han ayudado mucho a seguir creyendo en que si algo realmente te gusta, y luchas por ello, lo vas a conseguir.

Por supuesto no me olvido de mis grandes compañeros de universidad, en particular de aquellos que ahora forman parte de mis grandes amigos, Esparza, Gema y Adrián. Creo que sin ellos no hubiese sido posible recorrer todo este camino, y superar paso a paso cada nuevo reto.

Agradecerte también a ti, Fer, todo el apoyo que me has dado; quizá has sido quien más me ha aguantado con mis altibajos, y ha luchado junto a mí por conseguir esta meta. Muchas gracias por todo.

Y para terminar, me gustaría dedicar este TFG a alguien que hoy ya no está conmigo, y que sé que estaría muy feliz y orgullosa de verme finalizar mi carrera. Este TFG te lo dedico a ti, mamá.

# INDICE DE CONTENIDOS

# INDICE DE FIGURAS



# INDICE DE TABLAS

# 1 Introducción

## *1.1 Motivación*

En los últimos años, las tendencias de uso de Voz sobre IP (Voice over IP, VoIP) se han afianzado, produciéndose un aumento considerable de los despliegues y popularidad de esta tecnología. Esto ha supuesto un cambio disruptivo en el mercado, debido al potencial y las diferencias de este tipo de servicios de voz frente a la telefonía convencional. La revolución de la VoIP viene motivada por las ventajas que ofrece tanto a los proveedores de servicios como a los usuarios: menor coste, convergencia de servicios, flexibilidad y versatilidad y comodidad de uso.

El despliegue de servicios de telefonía sobre redes IP existe desde hace más de 20 años, como muestra la aparición del primer software con soporte para estos servicios (*VocalTec Internet Phone* programado por VocalTec en 1995) o los años de definición de algunos de los protocolos de señalización (por ejemplo, la recomendación H.323 de la ITU [2], aprobada por primera vez en 1996). No obstante, el despliegue comercial más allá de pruebas de concepto en laboratorios tardó algunos años más en adquirir relevancia: auge de estos sistemas a partir de los años 2000 se aprecia teniendo en cuenta el aumento de protocolos de VoIP (SIP o IAX) y del incremento tanto de prestadores de servicio como de usuarios de compañías y aplicaciones de telefonía sobre redes IP.

Tal es el nivel de éxito, que de 2005 a 2011 el número de usuarios de VoIP se incrementó en un 15%, lo que equivale a un aumento de más de 200 millones de usuarios (*Fuente: IDATE*), y que sigue creciendo actualmente, lo que hace que sean numerosas las empresas que apuestan por su utilización centrando sus investigaciones e inversiones en mejoras continuas y nuevas funcionalidades. También son numerosas las herramientas que permiten el uso de esta tecnología, cada vez más completas ofreciendo al usuario sencillez, a la vez que completitud en las opciones disponibles, facilitando el día a día en múltiples entornos, como pueden ser desde una simple llamada particular al seguimiento médico de una persona enferma sin tener que trasladarse al hospital, y en la mayoría de las ocasiones a coste cero al ser por conexión de banda ancha. Otro de las grandes utilidades de esta herramienta, es para las empresas, puesto que al canalizar sus comunicaciones a través de IP, tienen control total del tráfico de sus llamadas tanto internas como externas y posibilidad de extraer muchos datos que pueden ser críticos por ejemplo en un *Call Center*, como origen y destino, estado de la llamada, llamadas en espera o tiempo medio de llamadas.

Debido a este incremento de usuarios y el éxito de esta tecnología, surge la necesidad de la creación de nuevas herramientas para la monitorización y control del tráfico de este tipo de llamadas. Aunque existen numerosas herramientas de monitorización, hay gran cantidad de parámetros a medir que nos pueden ser muy útiles en la realización de mejoras y de correcciones que no todas éstas implementan. Además, el software existente normalmente está optimizado hacia un servicio o empresa concreta, de tal forma que los servicios que presta la herramienta, su coste total tanto de procesado como de recursos y su utilidad no es universal para cualquier usuario.

Como solución a esto, surge el proyecto VoIPCallMon [1], desarrollado por el grupo de investigación de Redes y Computación de Altas Prestaciones (*High Performance*

*Computing and Networking research group*, HPCN) de la Universidad Autónoma de Madrid, en el cual se idea un software que está optimizado en términos de coste computacional y consumo de recursos, (por lo que se programa en C) con el fin de permitir la monitorización de grandes despliegues de VoIP. Su diseño, modular y con minimización de las dependencias entre los distintos bloques funcionales, permite la extensión y adecuación del sistema a diferentes entornos de instalación, y se puede ejecutar sobre hardware de propósito general. De esta forma, se convierte en una solución flexible y que con un coste reducido proporciona una funcionalidad completa y de fácil integración en despliegues operacionales.

No obstante, el hecho de introducir servicios como VoIP sobre redes de comunicaciones, en gran medida basadas en la pila de protocolos TCP/IP, plantea una serie de complicaciones adicionales. En particular, existen ciertas diferencias relativas al aseguramiento de calidad de servicio y a la forma de funcionamiento de las redes telefónicas convencionales y la la VoIP.

A diferencia de las redes tradicionales de telefonía o PSTN basadas en la conmutación de circuitos, las redes IP se basan en la conmutación de paquetes. Esto es, que no se va a tener un circuito dedicado como en PSTN, si no que por un mismo enlace o nodo pasarán diferentes paquetes de diferentes servicios, y por lo tanto el tráfico de paquetes vendrá limitado por el ancho de banda de las líneas y nodos, además de mezclarse con muchos otros tipos de paquetes e información.

Por otro lado, el protocolo de la capa de transporte TCP, o *Transmission Control Protocol,* sobre IP, el cuál es uno de los protocolos fundamentales de internet, garantiza que el dato sea entregado, y en el orden en el que se ha transmitido sin errores, algo que IP no asegura. Pero en este caso, al ser una herramienta en tiempo real, los paquetes viajan usando el protocolo RTP o *Real-Time Transport Protocol,* a nivel de sesión, lo que implica que en la capa de transporte irán sobre UDP o *User Datagram Protocol* y no sobre TCP. Esto provoca que no se asegure la llegada de paquetes ni tampoco en el orden con el que se han transmitido [3]. Problemas en las redes como es el *jitter*, pérdidas, colapso en los anchos de banda y nodos, etc, hacen que los paquetes se pierdan, que lleguen con demasiado retraso como para ser útiles o que lleguen desordenados, lo que causa una pérdida de Calidad de Servicio (QoS) [4] que debe ser controlada en todo momento para asegurar un nivel mínimo de calidad.

Así, necesitamos herramientas que controlen la calidad de los servicios de voz desplegados. En el caso de este Trabajo de Fin de Grado, nos centraremos en el desarrollo e intergración de nuevos elementos funcionales que permitan obtener métricas para evaluar la calidad de servicio y experiencia que complementen las actualmente generadas por el sistema, y mejorar los valores proporcionados para otras métricas ya incluidas en los registros de conexiones asociadas a los servicios de VoIP deplegados en las redes monitorizadas.

## *1.2 Objetivos*

El principal objetivo de este TFG es la mejora y extensión del sistema VoIPCallMon, para que se puedan integrar en los procesos de monitorización acometidos con esta herramienta..

La idea de estas mejoras, surge de la necesidad de ir complementando la herramienta de tal forma que se pueda adaptar a cualquier entorno o usuario de VoIP que quiera monitorizar sus sistemas, y que sea lo más completa posible evitando que se necesiten herramientas adicionales y mayores desembolsos para medir todos los parámetros relevantes para asegurar la calidad de servicio y experiencia de los usuarios finales.

La idea de estas mejoras, surge de la necesidad de ir complementando la herramienta de tal forma que se pueda adaptar a cualquier entorno o usuario de Voz IP que quiera monitorizar sus sistemas, y que sea lo más completa posible evitando que se necesiten herramientas adicionales y mayores desembolsos para medir todos los parámetros que el usuario final desee.

De este modo, los objetivos concretos para este TFG son los siguientes:

- Implementación de medición de PDD (*Post Dial Delay*), de tal forma que este nuevo dato medido quede grabado en el archivo de salida para poder consultarse en cualquier momento.
- Conseguir adaptar la medición estadística de pérdidas lo máximo posible a las pérdidas reales que mediríamos en uno de los extremos de la conexión, implementando un buffer de tolerancia de pérdidas ajustado para que la medición sea lo más exacta posible.
- Valorar diferentes posibles herramientas de transcripción, para una vez realizada la grabación de la conversación poder obtener una transcripción completa en texto de la misma.

Con todos ellos, se conseguirá una mejora significativa en las funciones que tiene la herramienta. Además, las conclusiones y los principales resultados obtenidos junto con las mejoras de funcionalidad motivarán y permitirán definir nuevas líneas de trabajo futuro.

## *1.3 Organización de la memoria*

El resto de esta memoriase organiza de la siguiente forma:

- **En el Capítulo 2:** se desarrolla el estado del arte del TFG; se desarrolla el estado actual de la Voz IP, cómo surge esta técnica, el funcionamiento general de este tipo de redes y, por último, se desarrolla brevemente la herramienta VoIPCallMon sobre la que se realizarán las mejoras.
- **En el Capítulo 3:** Una vez visto el funcionamiento de las redes VoIP y de la herramienta VoIPCallMon, en este tercer capítulo se desarrollarán y enumerarán los requisitos de la aplicación y de las mejoras que se realizan en este TFG, dando una visión más técnica de dichas mejoras.
- **En el Capítulo 4:** Vistas las mejoras a realizar y sus requisitos, en este capítulo se desarrollará la implementación de las mismas en la herramienta.
- **En el Capítulo 5:** Con las mejoras ya implementadas, se realizan pruebas que corroboran que estas mejoras funcionan y que dan los resultados esperados, contrastando con resultados anteriores y/u otras herramientas.

- **En el Capítulo 6:** en este último capítulo se aportan las conclusiones a las que se llegan tras las mejoras realizadas así como las mejoras futuras que se podrían aportar a la herramienta.

## *1.4 Fases de Realización del Proyecto*

La implementación de las mejoras para VoIPCallMon, se llevará a cabo en las siguientes fases:

### 1.4.1 Estado del Arte y Documentación

En esta fase, se busca información y documentación sobre todo aquello que pueda ser relevante para este TFG y las mejoras que se van a implementar. Una vez documentado, se obtendrá una visión acerca del estado de desarrollo de herramientas similares a VoIPCallMon y los requisitos que deben satisfacer para mejorar la evaluación de calidad y las decisiones de gestión para despliegues de VoIP.

### 1.4.2 Diseño

En esta fase, se idea cómo van a ser las implementaciones de las mejoras a realizar, cuáles son sus requisitos, de forma explícita y comprobable, y en qué parte de la herramienta y cómo se van a desarrollar esas mejoras.

### 1.4.3 Desarrollo

Teniendo en mente el listado de requisitos que se han definido durante la fase de diseño, se pasa a la implementación de la mejora en la parte correspondiente del software, intentando optimizar esta implementación lo máximo posible y sin afectar al resto del código de la herramienta.

### 1.4.4 Integración, Pruebas y Resultados

Se comprueba que los requisitos inicialmente definidos han sido cubiertos completamente (lo que garantiza el correcto desarrollo e implementación de las mejoras) y se interpretan los resultados obtenidos con las pruebas.

### 1.4.5 Conclusiones

Una vez comprobados los resultados de las pruebas y pasadas las mismas satisfactoriamente, extraemos las principales conclusiones y lecciones aprendidas a partir de se analizan los datos obtenidos y los desarrollos realizados, sugiriendo futuras mejoras posibles que han ido surgiendo a lo largo del desarrollo de este TFG.

# 2 Estado del arte

## 2.1 Introducción

En esta sección, se explicarán diferentes conceptos y escenarios que son necesarios para la comprensión del resto del trabajo llevado a cabo. Así, se va a definir a qué se refiere el término VoIP, cuáles son sus funciones principales, y los elementos y protocolos habitualmente presentes en los despliegues de este tipo de servicios. Posteriormente, se presentará la herramienta VoIPCallMon, en la cual se han integrado los elementos funcionales desarrollados, detallando su funcionamiento y motivando de esta forma las mejoras que se realizan, su diseño e implementación.

## 2.2 VoIP (Voice over Internet Protocol)

El término VoIP, o también conocido como voz sobre protocolo de internet, es un grupo de recursos que permiten la transmisión de la voz a través de internet empleando para ello el protocolo IP. De esta forma, la señal de voz se envía en forma de paquetes a través de la red digitalmente, pasando por los diferentes nodos que los irán direccionando hasta su destino. Por este motivo, cuando hablamos de una red IP, hablamos de una red *best effort*, o mejor esfuerzo, puesto que a diferencia de las redes de telefonía convencional *PSTN*, no podemos asegurar la llegada de los paquetes [3]. Por ejemplo, si en uno de los nodos, se reciben más paquetes de los que puede admitir, se saturará y parte de éstos quedarán descartados no llegando a su destino final.

Otras de sus desventajas destacables es que existen diferentes protocolos de uso, y algunos de ellos son privados, lo que requiere que ambos usuarios para poder comunicarse dispongan de la misma herramienta, como puede ser por ejemplo el caso de Skype, el conocido software de Microsoft. También es cierto, que la seguridad que nos ofrece un protocolo privado como es el de Skype no nos lo puede ofrecer un protocolo abierto, puesto que es mucho más vulnerable a posibles escuchas malintencionadas con el propósito de obtener datos privados, por lo que se necesita cierto balance entre las funciones de la herramienta, la seguridad que ofrece, su coste, etc.

En cualquier caso, la voz sobre IP es mucho más flexible que la red convencional PSTN, pensada para transmisión de voz únicamente. Al realizarse a través de Internet, es mucho más versátil y permite el envío a la vez de vídeo, archivos, facilita la realización de multiconferencias y la recepción de múltiples llamadas en una sola línea (como es el caso de los *call centers*). Por otro lado, facilita la gestión de los servicios telefónicos ya que permite la convergencia de la obtención de datos y medidas a partir del tráfico de red mediante herramientas de monitorización para redes IP.

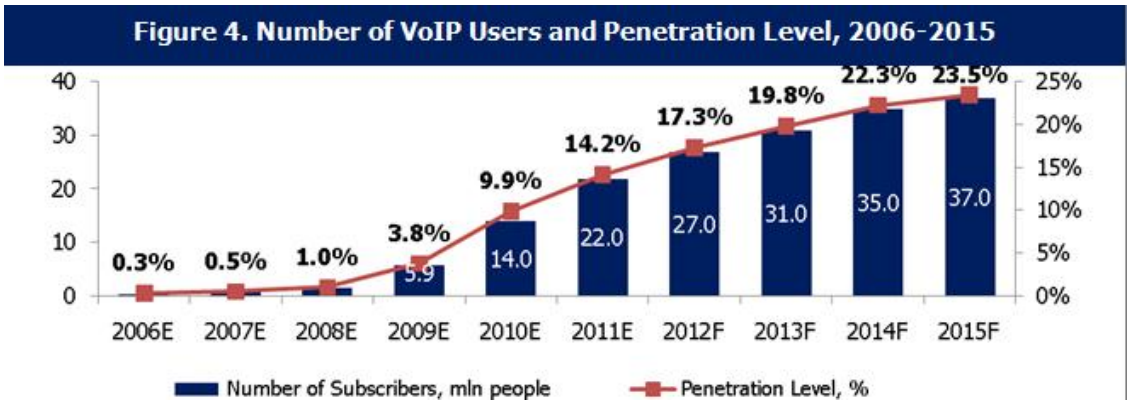


**Figura 2-1: Crecimiento Usuarios VoIP J'son & Partners Consulting, 2012**

Según datos de J'son & Partners, de 2006 a 2011 aumentó en 25 millones el número de usuarios de VoIP. Las estimaciones en ese momento pronosticaron además un crecimiento de un 10% adicional para 2015 , lo cual nos da una visión de la importancia que tiene la voz sobre IP actualmente en términos de uso. Año tras año se impone a la tradicional, puesto que estas estimaciones indican que los operadores están implantando infraestructuras de red para realizar las comunicaciones cada vez más con VoIP.

### 2.2.1 Red VoIP

Con el fin de conocer un poco mejor el funcionamiento de una red típica de VoIP y conocer los elementos que la forman, a continuación se describirán los principales elementos que se suelen encontrar en los despliegues de este servicio. El equipamiento de red implicado se puede dividir, según sus funciones dentro de la comunicación, en las siguientes categorías:

- **Terminales:** son los extremos de la comunicación. Aquí se engloban todos los dispositivos que un cliente usa en este tipo de comunicación, como altavoces, micrófonos, interfaces de usuario o teléfonos VoIP especialmente diseñados para este sistema. Aquí se establecen y originan las llamadas de voz, y los datos transcritos por el micrófono son enviados en forma de paquetes a la red.
- **PBX:** conocido también como servidor o *gatekeeper*. Registra los usuarios, sus direcciones dentro de la red IP, actúa como traductor entre dispositivos, controla el tráfico, etc. Una vez registrados los terminales y usuarios integrantes de la red, permite que sean localizados con mayor rapidez y facilidad.
- **Gateways:** permiten la interconexión y salida de la comunicación al resto de la red, como internet o con redes de telefonía tradicional PSTN, ya sea fija o móvil.

Para comprender mejor cómo se interconectan estos elementos y cuál es la topología de los despliegues de VoIP, a continuación se muestra un pequeño diagrama que muestra su estructura típica:

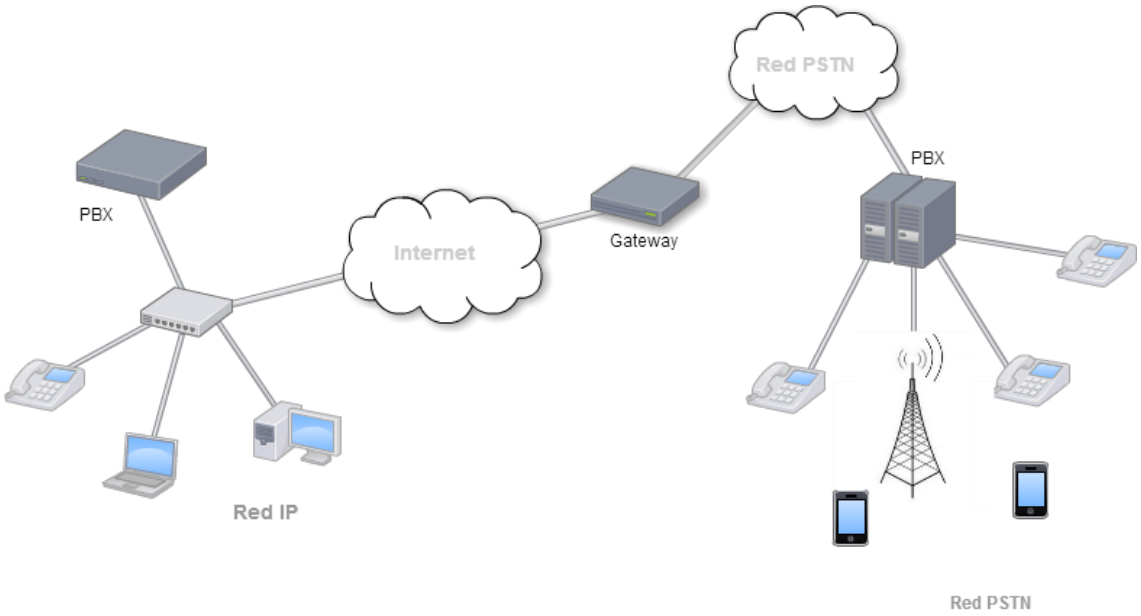


**Figura 2-2: Diagrama estructura típica red VoIP-PSTN**

En una estructura tipo a la descrita anteriormente, el proceso típico en una llamada sería el siguiente:

- **Verificación**: se verifica que el usuario puede realizar la llamada, es decir, que tiene saldo disponible en el caso de herramientas privadas, o que el otro usuario está disponible para poder realizar dicha llamada.
- **Negociación:** en esta fase se decidirá si se trata de llamada de voz o videollamada, el códec a usar en la misma, habitualmente se suele usar o el G.711 o el G.729, así como los puertos en los que se va a establecer la conexión.

Una vez negociados los parámetros anteriores, comienza la comunicación a través de RTP generalmente, con flujos de audio y/o video. Si uno de los usuarios envía un mensaje de finalización, la llamada acabará. Para ilustrar una llamada típica de VoIP realizada con el protocolo SIP y ejecutada correctamente, se muestra el siguiente diagrama que muestra el intercambio de mensajes tanto de señalización y negociación de la llamada como de los datos multimedia:

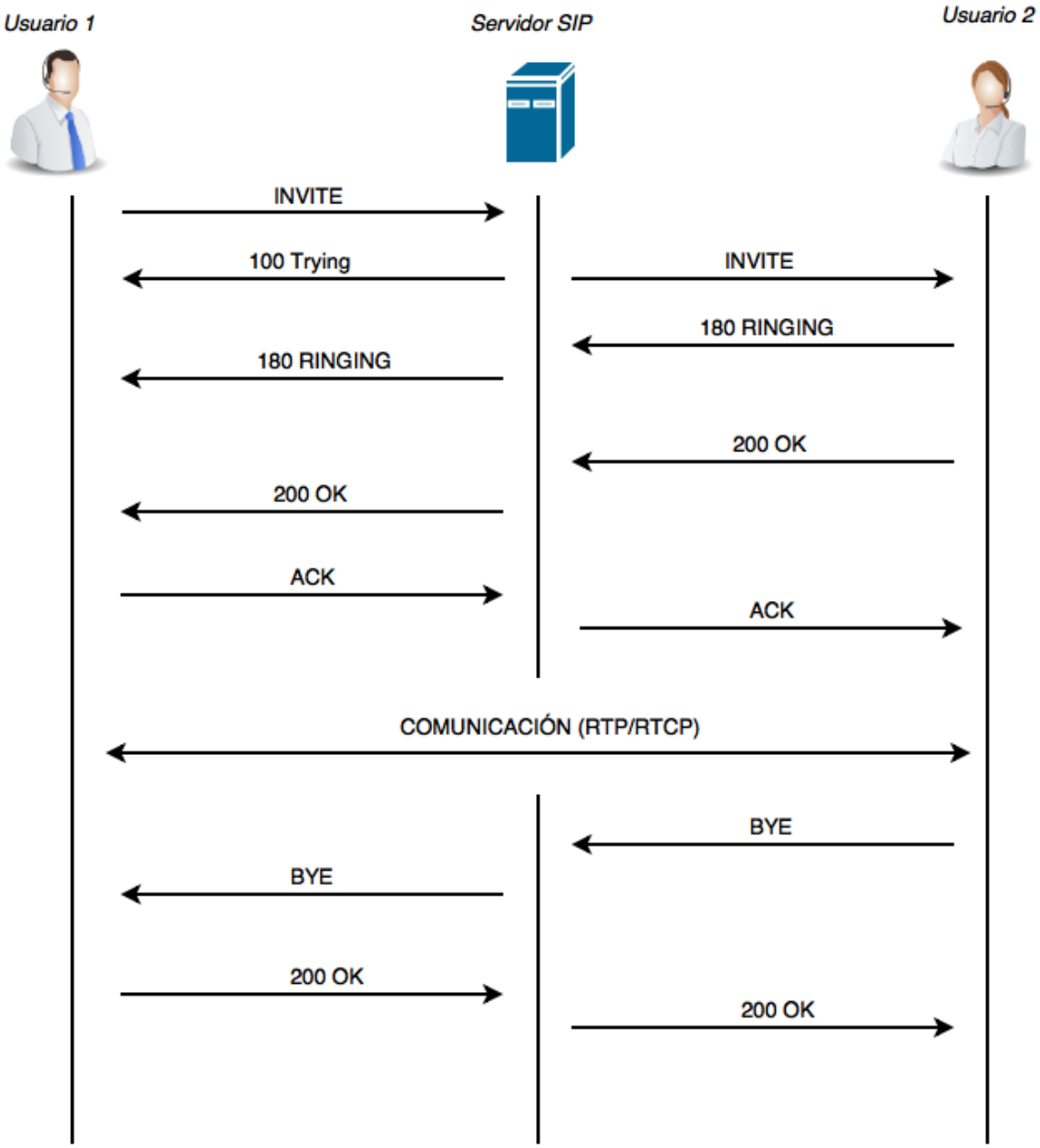


**Figura 2-3: Estructura de llamada VoIP**

En el diagrama se pueden observar los paquetes SIP intercambiados en las fases de Negociación y Verificación antes mencionadas, y después un flujo de paquetes RTP/RTCP que se intercambian directamente entre los usuarios, es decir, en una llamada VoIP podemos distinguir:

- Tráfico de señalización: son todos aquellos paquetes que van a través del servidor SIP y cuyos paquetes básicos serían INVITE, Trying, RINGING, OK, ACK, BYE. En el intercambio de estos paquetes se produce la verificación y negociación.
- Tráfico multimedia: son todos aquellos paquetes que contienen la voz y/o video en su caso, es decir, contienen la conversación. Estos paquetes viajan a través del protocolo RTP/RTCP y es una conexión directa entre ambos usuarios, no pasa por el servidor SIP.

Visto el funcionamiento típico de una red VoIP, a continuación se describe la herramienta de monitorización pasiva de tráfico de red *VoIPCallMon.*

### 2.2.2 VoIPCallMon

Las funcionalidades realizadas en este TFG se realizan sobre el proyecto VoIPCallMon, el cual se describirá su finalidad y funcionamiento en este apartado para dar paso así a las mejoras realizadas posteriormente con una mejor compresión.

El proyecto VoIPCallMon surge de la necesidad de monitorizar el tráfico de VoIP a causa de la expansión de su uso, y de resolver los problemas existentes en esta tecnología. Es necesario para grandes tráficos de llamadas, sobre todo en las grandes empresas que prestan este servicio, tener puntos de monitorización de la red, lo que conlleva una inversión grande para ello. Con este objetivo, nace VoIPCallMon, haciendo así que la monitorización sea mucho menos costosa en hardware pero con un software optimizado de alto rendimiento, teniendo así disponibles todos aquellos datos que nos pueden ser de utilidad o que son necesarios como la latencia, tasa de errores, etc.

A continuación, se explica la estructura del software.
VoIPCallMon está dividido en cuatro módulos funcionales bien diferenciados [1]:

- **Módulo de captura de tráfico:** su función es la de capturar y filtrar el tráfico que pasa por la red buscando sólo aquellos paquetes que pertenecen a VoIP en cualquiera de los protocolos soportados por la herramienta (SIP, SCCP, Unistim o RTP). Alcanza velocidades de captura entorno a los Gb/s con hardware común fácilmente adquirible en el mercado.
- **Módulo de detección y seguimiento de tráfico VoIP:** este módulo es el encargado de comprobar si una vez capturado el tráfico se trata efectivamente de un mensaje de protocolo VoIP, y si se trata de una nueva llamada o es una que ya está en curso. Posteriormente se buscarán en el mensaje otros datos de interés como la IP de origen y destino, puertos, etc. En este módulo será donde se introduzcan las mejoras realizadas en la herramienta en este TFG.
- **Módulo de generación de llamadas y grabaciones VoIP:** aquí es donde, una vez finalizada la llamada, se genera toda la información acerca de ésta, y también algunos parámetros para medir la QoS *(Quality of Service).*
- **Módulo de retención de llamadas VoIP:** periódicamente, este módulo se encarga de volcar toda la información extraída en el módulo anterior a una base de datos MySQL. De esta forma, la información queda guardada para ser accesible en cualquier otro momento y tener datos que pueden ser de gran utilidad posteriormente.

Dentro de estos cuatro módulos se engloban todas las funcionalidades de la herramienta, y en ellos se incluirán las mejoras a realizar en este TFG así como mejoras futuras. En el siguiente diagrama se muestra la estructura diferenciada por bloques, así como el emplazamiento de la herramienta VoIPCallMon en una red habitual. El funcionamiento de la misma sería dentro de una red corporativa, la herramienta captaría el tráfico, filtraría los paquetes que son de VoIP y usan alguno de los diferentes protocolos que puede admitir, lo analizaría extrayendo los datos y cálculos para los mismos, y lo enviaría de forma pasiva.

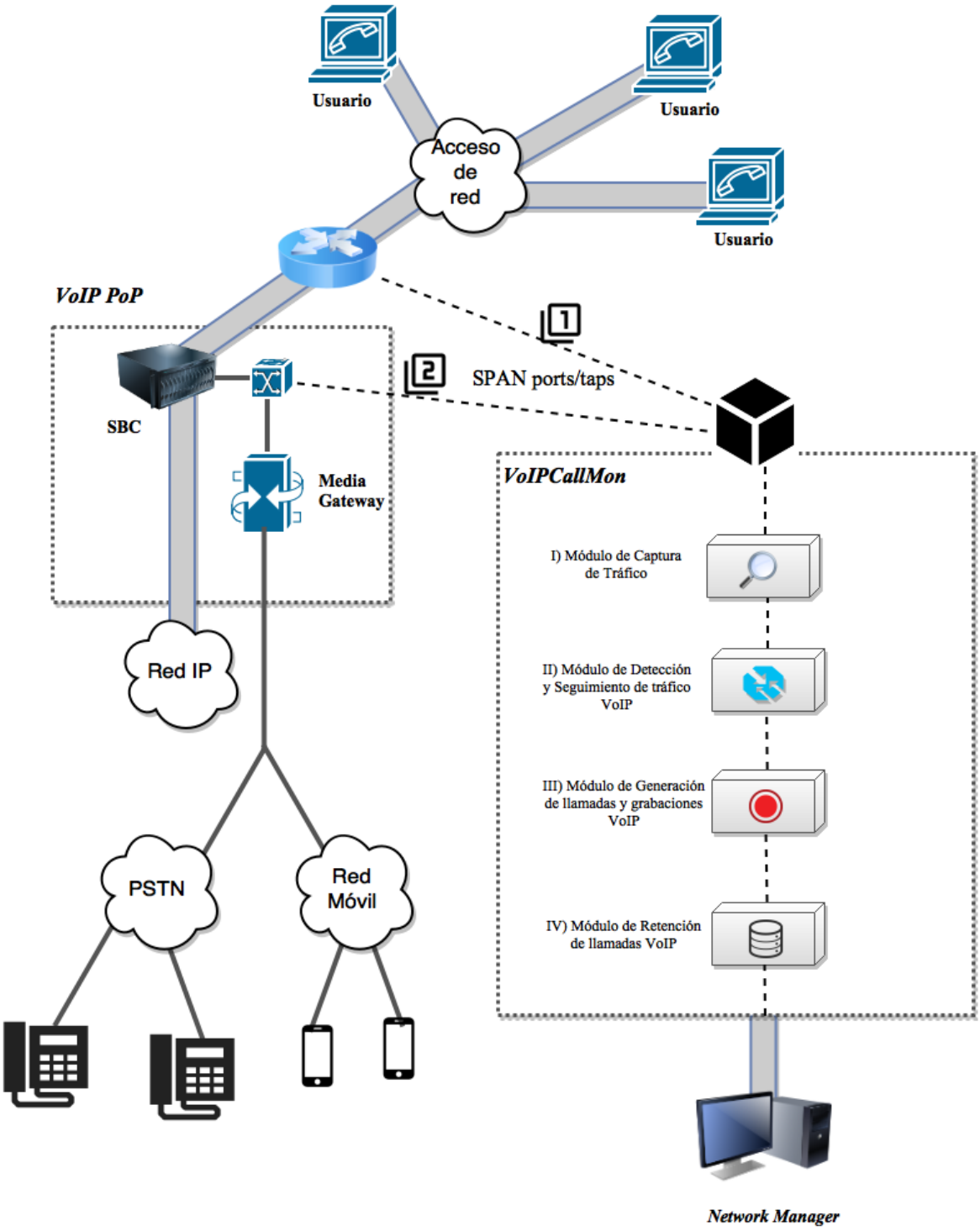


**Figura 2-4: Estructura VoIPCallMon**

## 2.3 Conclusión

En este apartado se ha desarrollado y presentado el estado del arte de la voz IP, las mejoras a implementar y descrito el funcionamiento de VoIPCallMon, en el cual se implementarán las mejoras a llevar a cabo en este TFG. En el siguiente capítulo se desarrolla el diseño que se va a llevar a cabo y los requisitos que ha de cumplir cada una de las mejoras que se van a llevar a cabo.

# 3 Diseño

## 3.1 Introducción

En este capítulo se desarrollarán las mejoras necesarias a llevar a cabo en este TFG y los requisitos necesarios para éstas.Para ello, partiendo de la descripción de la herramienta VoIPCallMon, analizaremos las mejoras que se van a llevar a cabo, que serán tal y como se ha comentado anteriormente la implementación de medición de PDD, la implementación de un buffer para mejorar la estimación de pérdidas, y por tanto del MOS, y la transcripción de las grabaciones de VoIP. De este modo, se conseguirá complementar la herramienta VoIPCallMon con nuevas funcionalidades que son realmente útiles en una red VoIP típica.

## 3.2 PDD

### 3.2.1 Definición

Post Dial Delay (PDD) [5] es, en VoIP, el tiempo que transcurre desde que el usuario origen envía el último tono de la numeración del usuario destino hasta que escucha el primer tono de llamada es decir, desde que se envía el primer paquete INVITE hasta recibir el primer paquete de RINGING. Este tiempo es importante tenerlo en cuenta puesto que cualquier usuario al realizar una llamada si pasado un tiempo no recibe tono de llamada, colgará suponiendo que la llamada no ha podido realizarse por algún motivo, por lo que este debe ser lo menor posible.

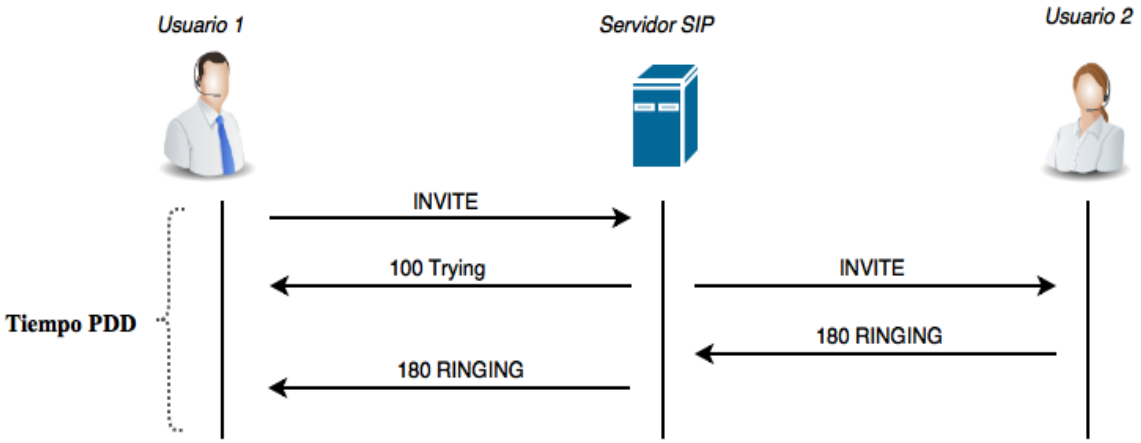


**Figura 3-1: PDD**

### 3.2.2 Requisitos

Para realizar esta mejora, se debe tener en cuenta que:

- Captar los *timestamp* del primer INVITE y RINGING de la llamada.
- Las unidades en las que están tomados los *timestamp* de los paquetes
- Realizar la resta de ambos para obtener el cálculo del PDD

- Guardar el cálculo realizado y sacar el mismo al archivo con todos los datos de la llamada.
- Comprobar que se está midiendo correctamente.

Este tiempo, debe ser lo más pequeño posible y es de vital importancia que se tenga en cuenta en una herramienta de monitorización pasiva como es VoIPCallMon, ya que si el usuario que llama no recibe respuesta tras marcar la numeración en un tiempo grande, considerará que no se está realizando la conexión correctamente y abandonará la llamada.

## *3.3 Buffer de tolerancia de pérdidas*

Las pérdidas se deben controlar en todo momento en una red de VoIP, ya que pueden hacer que la llamada sea ininteligible y que la calidad de experiencia percibida por los usuarios decaiga drásticamente. La mejora introducida será en el mismo módulo que el PDD citado anteriormente, es decir, en el de detección y seguimiento de tráfico.

### 3.3.1 Definición

Anteriormente, en la herramienta solo se realizaba una estimación estadística de pérdidas. Al tratarse de una herramienta de monitorización pasiva, esta estimación a veces puede alejarse bastante de la realidad, puesto que no tenemos información sobre la tolerancia que hay en los extremos de la conexión a posibles pérdidas, en la que habitualmente un buffer hace que no por una mínima variación en el tiempo de llegada entre paquetes o si un paquete llega descolocado se considere directamente una pérdida. A continuación se muestra un diagrama que ayudará a entender mejor el funcionamiento de un buffer de pérdidas.

- Si los paquetes llegan desordenados:

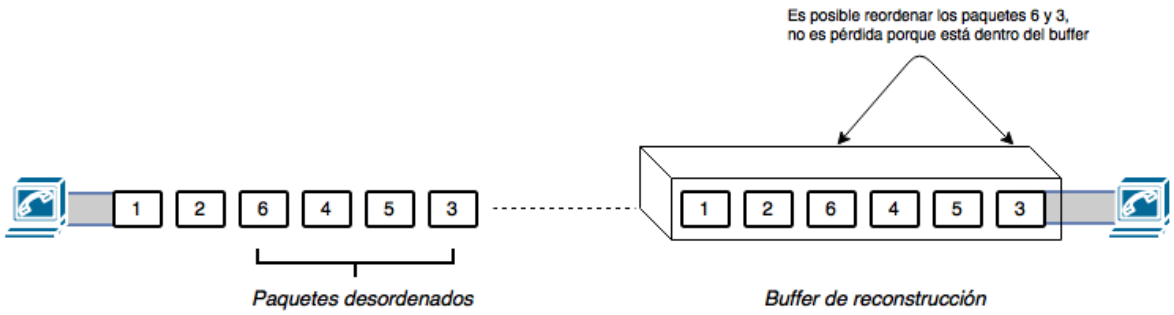


**Figura 3-2: Diagrama de paquetes desordenados en buffer**

- Si el tiempo entre llegadas aumentase:

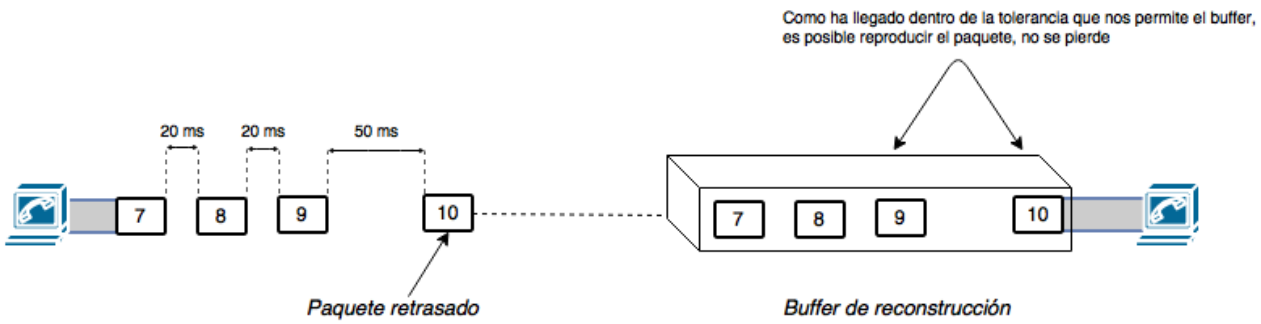


**Figura 3-3: Diagrama de paquetes con retardo en buffer**

Este buffer tendrá un tamaño determinado, y nos determinará en el caso de los números de secuencia cuántos números de diferencia podemos tolerar a partir del tiempo de paquetización del códec que se está usando. A continuación se muestran los tiempos de muestreo y el tiempo de paquetización de algunos de los códec que se suelen usar [*Fuente: Cisco*].

| Códec | Códec Sample Interval (ms) | Voice Payload Size (ms) |
|---|---|---|
| **G.711** | 10 ms | 20 ms |
| **G. 729** | 10 ms | 20ms |
| **G. 723.1** | 30 ms | 30 ms |
| **G. 726** | 5 ms | 20 ms |
| **G. 728** | 5 ms | 30 ms |
| **G.722_64k** | 10 ms | 20 ms |

**Tabla 1: Códecs más utilizados**

Como los códec habituales son G.711 o G.729 y tienen los mismos tiempos de paquetización, en este caso el buffer se ajusta de igual forma para ambos sin necesidad de comprobación. Si por ejemplo el buffer que queremos es de 200ms, al dividirlo entre 20ms de paquetización obtenemos que en nuestro buffer se toleran hasta 10 paquetes, es decir, 10 números de secuencia.

### 3.3.2 Requisitos

Para poder ajustar este buffer, como las herramientas que usa cada usuario pueden variar de unas llamadas a otras, el buffer que usan estas herramientas será diferente, e incluso personalizable. Por ello, se requiere de la realización de pruebas estimando el *Mean Opinion Score* (MOS), opinión de la calidad de la red por parte del usuario en un valor entre 1 y 5 [3], que en VoIPCallMon se realiza basándose según las directrices de la ITU en su recomendación G.107 [6] , en el cual se proporciona el método E-Model para relacionar calidad medidas del nivel de transporte. Además, se sigue la adaptación de servicios de VoIP que se presentan en *Cole and Rosenbluth* [7], el cual usa una serie de parámetros relacionados con el códec usado específicamente, RTT, *jitter* y las pérdidas cuantificadas. Así, los requisitos son:

- Hacer una primera aproximación con tolerancia de números de secuencia no consecutivos y obtener resultados.
- El buffer debe tener en cuenta el tiempo de paquetización de los códec utilizados (en este caso 20ms, códec G.711 y G.729).
- Comprobar que el MOS está variando con el cambio anterior, con diferentes trazas y retardos.
- Realizar un segundo cambio con tolerancia tanto a números de secuencia no consecutivos como a retardos mayores de lo normal para los códec usados.
- Comprobar de nuevo que el MOS varía con diferentes trazas y retardos.
- Comprobar con diferentes tamaños de buffer que mejoran los resultados
- Contrastar resultados con otra herramienta del estado del arte: aquí se realizarán tablas comparativas con cálculos de media y desviación típica para poder comprobar de una forma más sencilla que la nueva implementación está funcionando y que los resultados son los esperados.

## *3.4 Transcripción de las grabaciones*

### 3.4.1 Definición

Con el fin de que una conversación sea accesible a cualquier tipo de persona, inclusive con discapacidad auditiva por ejemplo, y también para poder tratar y archivar las conversaciones de la forma más sencilla y menos costosa posible, es importante tener una transcripción en texto de la conversación. Por este motivo, se deben valorar diferentes alternativas como herramienta de transcripción, ya que implementar una nueva sería muy costoso y conllevaría mucho tiempo. El objetivo final es que VoIPCallMon en su módulo de grabación de las conversaciones, transcriba las mismas creando un archivo de texto y se almacene.

### 3.4.2 Requisitos

En este caso, los requisitos son más estrictos. Se debe tener en cuenta lo siguiente:

- Transcripción local o de forma remota: preferiblemente la transcripción ha de realizarse de forma local, ya que necesitamos que la transcripción se realice de la forma más segura posible y evitar que la conversación tenga que viajar por la red, ya que los datos son confidenciales e internet no es seguro 100%.
- Coste de la herramienta lo más bajo posible: se necesita una herramienta que no sea muy costosa y si es posible gratuita, puesto que la idea del proyecto VoIPCallMon es una herramienta con bajo coste tanto de recursos Hardware como de Software, y que se pueda utilizar en cualquier equipo.
- Transcripción a archivo de texto: necesitamos un archivo de texto con la transcripción para poder trabajar con él y almacenarlo.
- Sistemas Operativos en los que funciona la herramienta: la herramienta debe ser universal y trabajar con el mayor número de Sistemas Operativos posible, pero obligatoriamente con sistemas GNU/Linux, ya que es el sistema en el que se ha implementado VoIPCallMon.
- Funcionalidad de la herramienta: el software deberá poder transcribir, a partir de las grabaciones, a texto en un archivo para su posterior almacenamiento y/o

tratamiento. Puede haber otras funciones de utilidad como pueden ser la posibilidad de varios formatos de salida en el archivo

- Código abierto o privado: es interesante la opción de una herramienta de código abierto ya que nos permitiría la posibilidad de incluirlo en el código de VoIPCallMon o poder modificarlo añadiendo funciones que puedan ser de interés para el proyecto.

## *3.5 Conclusión*

Una vez vistas todas las mejoras que se quieren realizar a nivel de diseño y sus requisitos a lo largo de este capítulo, se tiene una idea más clara de lo que se quiere realizar y cómo para después poder implementarlo de forma específica. Tras esta descripción de los requisitos de diseño de las mejoras, se pasará a explicar la implementación de las nuevas funcionalidades, de las que dotaremos a la herramienta, y los pasos seguidos para ello.

# 4 Desarrollo

## *4.1 Introducción*

En este capítulo se explicará el desarrollo llevado a cabo para implementar las mejoras propuestas para VoIPCallMon descritas anteriormente. Se indicará como se ha llegado a esa solución, los procesos de desarrollo y las pruebas realizadas de su correcto funcionamiento. Dividiremos la explicación conforme a las mejoras realizadas, es decir:

- PDD
- Buffer de tolerancia a pérdidas
- Transcripción de las grabaciones

## *4.2 PDD*

### 4.2.1 Implementación

Para implementar esta mejora, en el módulo de detección y seguimiento de tráfico VoIP, siguiendo el mismo orden que en los requisitos descritos en el Apartado 3.2.2 modificamos el código de tal forma que cuando hay una llamada nueva y nos llega el primer paquete INVITE captamos su *timestamp*, al igual que cuando llega el primer paquete de RINGING, para que una vez tenemos los dos, haciendo la resta de los *timestamp* obtener el tiempo de PDD. Como ambos timestamp están en las mismas unidades (microsegundos) no es necesario hacer cambios de unidades. Una vez se ha implementado y guardado el PDD, se implementará también la salida del resultado en el archivo que genera VoIPCallMon con todos los datos acerca de la llamada. Posteriormente se explicará la forma de comprobar que esta mejora está operativa y lo hace de forma correcta. A continuación se muestra en pseudocódigo cómo se ha implementado esta mejora de la forma más simple posible.

```
FUNCIÓN 'ACTUALIZAR INFORMACION LLAMADA' (…)
{
(…)

SI ('Paquete Recibido'=='PRIMER_INVITE') ENTONCES

      'TimestampINVITE'=TimestampPaqueteActual;

SI NO, SI ('Paquete Recibido'=='PRIMER_RINGING')) ENTONCES {

      'TimestampRINGING'=TimestampPaqueteActual;
      PDD='TimestampRINGING'-'TimestampINVITE'; }
(…)
}
```

## *4.3 Buffer de tolerancia a pérdidas*

### 4.3.1 Implementación

Del mismo modo que en el caso del PDD, siguiendo el orden de requisitos descritos en el Apartado 3.3.2, primero, se implementará un buffer fijo, de tal forma que se de tolerancia a pérdidas a través de los números de secuencia. Si llega un paquete cuyo número de secuencia es mayor que el anterior más 'n+1' o menor que el anterior menos 'n', siendo 'n' un número entero que será del tamaño del buffer dividido entre el tiempo de paquetización, entonces se considera pérdida. Se realizan comprobaciones para ajustar el margen y ver que funciona correctamente que se detallan en el capítulo siguiente, y como este buffer ha de tener en cuenta también los tiempos de llegada entre paquetes como se ha explicado en el capítulo 3, posteriormente se implementará esta segunda parte de la mejora.
Para ello, veremos si el paquete anterior y el que acabamos de recibir tienen una diferencia entre llegadas mayor que 'm' milisegundos, que será variable en función del tamaño de buffer que establezcamos al igual que 'n' para el número de secuencia. Si esto es así, el paquete se contabilizará como pérdida. Aquí se ha de tener en cuenta que no todos los códec utilizan los mismos tiempos de paquetización, pero en este caso, los códec más utilizados que son G.711 y G.729, utilizan el mismo tiempo de paquetización, típicamente 20ms, por lo que queda como posible trabajo futuro la detección de otros tiempos de paquetización y la adaptación a códecs con menor frecuencia de aparición en despliegues de VoIP, ajustando dinámicamente el buffer para adaptarse mejor a ellos. A continuación se muestra en pseudocódigo una versión simplificada de la implementación realizada.

```
FUNCIÓN 'PÉRDIDAS' (…)
{
(…)

SI (('NúmeroSeqPaqueteActual'<('NúmeroSeqPaqueteAnterior'-[n])) Ó
('NúmeroSeqPaqueteActual'>('NúmeroSeqPaqueteAnterior'+[n+1]))
ENTONCES

     ContadorPerdidas+=1;

SI NO, SI (('TimestampPaqueteActual'-
'TimestampPaqueteAnteior')>[m]) ENTONCES

     ContadorPerdidas+=1;

(…)
}
```

## *4.4 Transcripción de las grabaciones*

### 4.4.1 Implementación

Una utilidad a tener en cuenta que podría ser imprescindible es la posibilidad de transcribir una conversación o grabación de voz sobre IP. Esta importancia se debe a múltiples motivos: por ejemplo las personas con discapacidades sensoriales, ya que tener el texto de una conversación puede ayudarles a comunicarse de una forma mucho más sencilla, o la

posibilidad de reducir el peso de un archivo de voz a un simple archivo de texto que una empresa de *call centers* podría enviar a sus clientes o departamentos, reduciendo de este modo el coste, tiempo y complejidad que supondría el análisis de un archivo de audio en las tareas de evaluación de la calidad de atención.
Existen múltiples librerías para dar soporte a esta funcionalidad, y tal y como se ha indicado en las secciones anteriores, el desarrollo de un software propio queda fuera del alcance del presente trabajo. Por ello, y con el fin de aumentar la funcionalidad de nuestra herramienta, a continuación se catalogan algunas de ellas en una tabla comparativa con las características principales desde el punto de vista del desarrollo de nuestro proyecto. En particular, las dimensiones de comparación que hemos seleccionado son los siguientes:

- Coste: puesto que la idea del proyecto VoIPCallMon es una herramienta de bajo coste, un software de transcripción que suponga un desembolso grande no se ajustaría a las bases de esta herramienta.
- Código abierto: será deseable una herramienta de código abierto para poder incluir dicho código en el de VoIPCallMon y, si fuese necesario, añadir funciones extra.
- Sistema Operativo: debe funcionar mínimo en sistemas GNU/Linux tal y como se ha descrito en los requisitos, ya que es el sistema en el que está implementado VoIPCallMon.
- Transcripción local o no: deberá ser local por privacidad y seguridad de los usuarios de VoIPCallMon.
- Funcionalidad: es requisito imprescindible que la herramienta transcriba las conversaciones a partir del archivo de voz, y que lo haga en un archivo de texto para su posterior almacenamiento y/o tratamiento.

En las siguientes tablas se muestra la comparativa realizada de las herramientas de transcripción siguientes:

| Herramienta | Gratuita | Código abierto | SO | Website |
|---|---|---|---|---|
| **Vox Forge** | ✓ | ✓ | **Todos** | **http://www.voxforge.org/es/about** |
| **Dragon Transcription** | ✗ | ✗ | **Windows y OS X** | **http://www.nuance.com/dragon/transcription-solutions/index.htm** |
| **OpenEars** | ✓ | ✗ | **iOS** | **http://www.politepix.com/** |
| **HTK** | ✓ | ✓ | **Todos** | **http://htk.eng.cam.ac.uk/** |
| **ISIP** | ✓ | ✓ | **Linux** | **https://www.isip.piconepress.com/projects/speech/** |
| **Julius** | ✓ | ✓ | **Linux y Windows** | **http://julius.osdn.jp/en_index.php** |
| **Google Speech** | ✗ | ✗ | **Todos** | **https://cloud.google.com/speech/** |
| **CMUSPhinx** | ✓ | ✓ | **Todos** | **http://cmusphinx.sourceforge.net/** |

**Tabla 2: Herramientas de transcripción del estado del arte analizadas, características**

| Herramienta | Transcripción local | Funciones | Compatibilidad con VoIPCallMon | Motivo de descarte |
|---|---|---|---|---|
| **Vox Forge** | ✓ | **Transcripción, dictado.** | ✓ | **No se hace localmente la transcripción y está en desarrollo** |
| **Dragon Transcription** | ✓ | **Transcripción en PC y en smartphone** | ✗ | **No es gratuita** |
| **OpenEars** | ✓ | **Transcripción** | ✓ | **Sólo está disponible para iOS** |
| **HTK** | ✓ | **Reconocimiento de voz** | ✓ | **Reconoce voz humana, no transcribe** |
| **ISIP** | ✓ | **Reconocimiento de Voz** | ✓ | **Proyecto obsoleto** |
| **Julius** | ✓ | **Transcripción** | ✓ | **Optimizada en Japonés** |
| **Google Speech** | ✗ | **Transcripción** | ✗ | **No es gratuita ni realiza la transcripción de forma local** |
| **CMUSPhinx** | ✓ | **Transcripción** | ✓ | **Fase pre-alpha** |

**Tabla 3: Herramientas de transcripción del estado del arte analizadas, compatibilidad y funcionalidad**

Se puede observar que la mayoría son gratuitas y que la transcripción es de forma local, que son quizá las dos características más valorables. Acerca de los algoritmos que usan, no se aporta gran información por parte de sus desarrolladores, pero sí que se sabe que utilizan modelos de voz para entrenar la herramienta.

Uno de los inconvenientes mayores que tienen este tipo de herramientas, es que todavía no funcionan de forma correcta siempre, y suelen producirse errores en la transcripción que a veces pueden ser vitales, puesto que un simple 'no' que se transcriba como 'sí' o viceversa puede hacer que se realice una contratación, por ejemplo, de un producto no solicitado. Por supuesto, siempre quedaría el respaldo de la grabación de voz en el caso de que se necesitase contrastar la información registrada con el fin de subsanar posibles errores, aunque limitaría la aplicabilidad de la herramienta en ciertos tipos de análisis automáticos. En cualquier caso, es difícil tener una herramienta de este tipo que no cometa errores, puesto que aunque si bien es cierto que la voz humana sigue bastantes patrones, a la hora de hablar una misma lengua, cada persona pronuncia de una forma, el ruido que hay en la llamada varía también de forma considerable en las diferentes llamadas, etc.

En el caso de VoIPCallMon, la instalación de esta nueva funcionalidad se realizaría de tal forma que los datos de audio extraídos por el módulo que realiza esta función, fuesen analizados por un nuevo módulo que generaría el archivo de texto a almacenar para una futura consulta.
Como la idea de VoIPCallMon es que sea una herramienta de bajo coste, capaz de coexistir con otras herramientas de análisis y ser desplegada en hardware de propósito general diverso, es necesario controlar la exigencias tanto computacionales como de recursos de los elementos que se encargarían de la parte de transcripción. De este modo, la

herramienta más adecuada debería ser una de código abierto con el menor coste económico y que no tenga como requisitos mucha potencia de CPU, para poder seguir manteniendo el modelo de explotación del sistema. Dentro de esto, además, puesto que el software funcionaría en sistemas GNU/Linux, la herramienta de transcripción debe cumplir también con este requisito y poder funcionar como mínimo en estos sistemas. Si a esto añadimos los requisitos de seguridad asociados a las características de los datos personales y, potencialmente, con restricciones de confidencialidad incluidos en las conversaciones, se hace necesario que la transcripción se realice de forma local.

## *4.5 Conclusión*

En este apartado se ha descrito y mostrado la implementación realizada en las mejoras para VoIPCallMon de PDD y de buffer de tolerancia a pérdidas. En el caso de VoIPCallMon, se han mostrado las características principales de cada herramienta, valorando si se cumplen los requisitos necesarios descritos en el diseño. A continuación, se desarrollan las pruebas realizadas para comprobar el correcto funcionamiento de las mejoras realizadas y que los resultados son los esperados, así como la valoración final de las herramientas de transcripción.

# 5 Integración, pruebas y resultados

## 5.1 Introducción

Una vez estudiadas las mejoras a llevar a cabo en la herramienta, y desarrollada su implementación, se comprueba que efectivamente la implementación es correcta. Para ello se realizan una serie de pruebas, dependiendo de la mejora, con el fin de asegurar que la mejora está funcionando bien y los resultados son los esperados.

## 5.2 PDD

Para la comprobación del PDD, primero con una traza concreta, de forma aislada en un entorno controlado, se comprueba cuál es su PDD, a través de Wireshark. Una vez se ha comprobado cuál debe ser el resultado esperado, lanzamos VoIPCallMon con la mejora implementada en el código, y la misma traza que ya tenemos medida, para corroborar el resultado de PDD esperado. Comprobado que es correcto, se pasa a realizar la misma medición en otras trazas diferentes para asegurarlo y se implementa salida de este dato en el archivo de resultados de la traza que crea la herramienta, para que quede registrado el PDD en la información de las conversaciones.

Un ejemplo de comprobación de PDD:

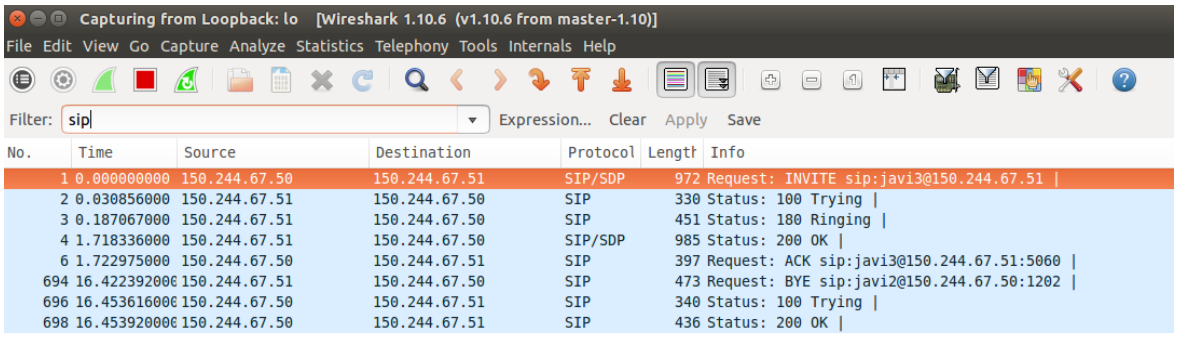


**Figura 5-1: PDD Wireshark**

El paquete No.1 es el INVITE y el paquete No.3 es el RINGING por lo que el PDD en este caso es de 0.187067 segundos. Si miramos el mismo resultado analizado con VoIPCallMon a través de la mejora implementada:

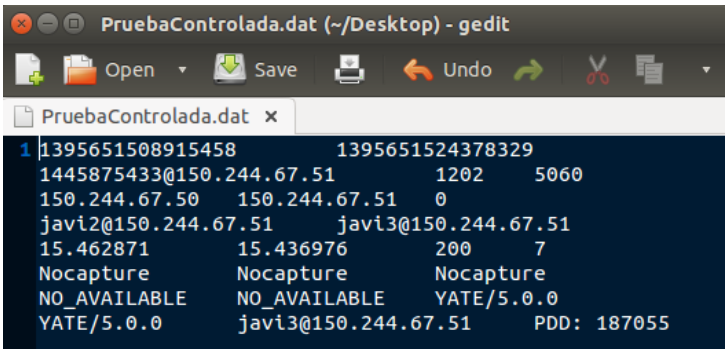


**Figura 5-2: PDD VoIPCallMon**

El resultado en este caso es de 0.187055 segundos, por lo que efectivamente el cálculo se está realizando bien con una variación en el PDD despreciable.

### *5.3 Buffer de tolerancia de pérdidas*

En esta mejora, es más difícil la comprobación, puesto que no conocemos el buffer real que puede haber en los extremos y además éste será variable. Por otro lado, se puede comprobar con el MOS, pero hacerlo con las opiniones reales de usuarios como rigurosamente se mediría el MOS es inviable debido a la implicación de muchos usuarios que conlleva y el coste que tendría. Por ello, en este caso, lo que se hace para realizar las pruebas es monitorizar con el sistema estadístico ya implementado anteriormente para el cálculo del MOS a partir de las pérdidas, obteniendo diferentes medidas del mismo con varias trazas y retardos, y con buffer variables. Primeramente, teniendo en cuenta sólo los números de secuencia, y una vez comprobado que la mejora funciona correctamente teniendo en cuenta también el tiempo de llegada entre paquetes.
Para llevar a cabo las medidas, y poder asegurarnos de que son fiables, se realizan 50 pruebas por retardo, a través de scripts que automatizan tanto la creación de estas trazas como la ejecución de VoIPCallMon y el tratamiento de los resultados, así como también el análisis de estas mismas trazas en VoIPMonitor [*Véase Anexo A*].
Se realizan medidas con retardos de 0ms, 5ms, 10ms, 20ms, 50ms, 75ms y 100ms. Estos retardos son introducidos en la red a través de la herramienta *netem* [10] de Linux.
Una vez se han realizado todas las mediciones se obtiene el MOS medio con cada retardo y su desviación estándar.
Para poder contrastar estos resultados, a través de otra herramienta similar a VoIPCallMon del estado del arte, como es VoIPMonitor [13], se termina de ajustar el buffer para que los resultados de MOS sean lo más similares posibles a los reales.
En cualquier caso, la estimación nunca será exacta, ya que solo analizamos el tráfico entre emisor y receptor, y en los extremos el usuario puede usar aplicaciones con tolerancias diferentes a las nuestras.

A continuación se muestran contrastadas las medidas de MOS antes y después de la implementación, comparadas con las medidas de MOS de la herramienta VoIPMonitor, las cuales mejoran considerablemente tras la implementación de dicho buffer.

- MOS con buffer de 50ms en VoIPMonitor y sin buffer en VoIPCallMon:

| Retardo | MOS medio VoIPMonitor | MOS medio VoIPCallMon | Diferencia Medias | Desviación típica VoIPMonitor | Desviación típica VoIPCallMon |
|---|---|---|---|---|---|
| 0 ms | 3,59 | 4,19 | 0,60 | 0,212506903 | 2,69159E-15 |
| 5 ms | 3,42 | 1,73 | 1,69 | 0,131800715 | 0,156669775 |
| 10 ms | 3,30 | 1,31 | 1,98 | 0,115987332 | 0,072324256 |
| 20 ms | 3,05 | 1,22 | 1,83 | 0,111098412 | 0,062308843 |
| 50 ms | 2,82 | 1,02 | 1,80 | 0,232370218 | 0,018433278 |
| 75 ms | 2,73 | 1,00 | 1,73 | 0,055106576 | 0 |
| 100 ms | 2,70 | 1,00 | 1,70 | 0,028571429 | 0 |

**Tabla 4: Comparación de MOS con buffer 50ms en VoIPMonitor y sin buffer en VoIPCallMon**

- MOS con buffer de 50ms en ambas herramientas:

| Retardo | MOS medio VoIPMonitor | MOS Medio VoIPCallMon | Diferencia Medias | Desviación típica VoIPMonitor | Desviación típica VoIPCallMon |
|---|---|---|---|---|---|
| 0 ms | 3,59 | 4,19 | 0,60 | 0,212506903 | 2,69159E-15 |
| 5 ms | 3,42 | 4,19 | 0,77 | 0,131800715 | 2,69159E-15 |
| 10 ms | 3,30 | 4,19 | 0,89 | 0,115987332 | 2,69159E-15 |
| 20 ms | 3,05 | 4,19 | 1,14 | 0,111098412 | 2,69159E-15 |
| 50 ms | 2,82 | 4,19 | 1,37 | 0,232370218 | 2,69159E-15 |
| 75 ms | 2,73 | 4,19 | 1,46 | 0,055106576 | 2,69159E-15 |
| 100 ms | 2,70 | 4,05 | 1,35 | 0,028571429 | 0,239227311 |

**Tabla 5: Comparación de MOS con buffer 50ms en ambas herramientas**

Vemos que la diferencia se ha reducido. Si comparamos el buffer de 200ms de VoIPCallMon con el de 50ms de VoIPMonitor, es cuando obtenemos los resultados más similares.

- MOS con buffer de 200ms en VoIPCallMon y de 50ms en VoIPMonitor

| Retardo | MOS medio VoIPMonitor | MOS Medio VoIPCallMon | Diferencia Medias | Desviación típica VoIPMonitor | Desviación típica VoIPCallMon |
|---|---|---|---|---|---|
| 0 ms | 4,01 | 4,19 | 0,18 | 0,231596059 | 2,69159E-15 |
| 5 ms | 3,89 | 4,19 | 0,30 | 0,122291055 | 2,69159E-15 |
| 10 ms | 3,70 | 4,19 | 0,49 | 0,160979147 | 2,69159E-15 |
| 20 ms | 3,60 | 4,19 | 0,59 | 0,141363621 | 2,69159E-15 |
| 50 ms | 3,30 | 4,19 | 0,89 | 0,266105702 | 2,69159E-15 |
| 75 ms | 3,08 | 4,19 | 1,11 | 0,168656153 | 2,69159E-15 |
| 100 ms | 2,89 | 4,05 | 1,16 | 0,137633053 | 0,239227311 |

**Tabla 6: Comparación de MOS con buffer 200ms en VoIPCallMon y de 50ms en VoIPMonitor**

Se observa que la diferencia en este caso es mucho menor. Esto puede ser debido a que no se sabe qué criterio estadístico está usando VoIPMonitor ni tampoco se sabe cuál es la mejor forma de calcular el MOS, puesto que no va a ser nunca 100% exacto sino una aproximación estadística. Pero lo que sí está claro es que la tolerancia a pérdidas con el buffer implementado está funcionando y mejora el MOS por ello.

## *5.4 Transcripción de las grabaciones*

Teniendo en cuenta los factores que se han evaluado anteriormente, factores, a continuación se desarrolla un análisis de las herramientas evaluadas:

- **VoxForge**: se trata de una herramienta pensada para realizar transcripciones a texto y ser implementada en otros Software de reconocimiento de voz como pueden ser HTK, ISIP, o Julius, pero aún está en desarrollo y en su web indican que sólo trabaja con algunos modelos acústicos y por tanto es necesario para poder usar la herramienta entrenarla con más modelos acústicos para el idioma que se vaya a utilizar y diferentes voces, por lo que necesitaríamos encontrar una gran variedad de modelos acústicos a fin de entrenar la herramienta. Por ello, aunque la herramienta podría ajustarse perfectamente a lo que necesita VoIPCallMon, habría que esperar a que esté totalmente disponible para al menos dos idiomas y que no sea necesario su entrenamiento continuo para evitar errores constantes si no que se pueda incluir en el proyecto directamente con un correcto funcionamiento lo más fiable posible.
- **Dragon Transcription:** esta herramienta aunque completa y con desarrollo terminado, se trata de una herramienta que no es gratuita, por lo que no encaja con las bases del proyecto VoIPCallMon, ya que esto supondría un desembolso periódico para poder mantener operativa esta función.
- **OpenEars:** es aplicación móvil diseñada para iOS, por lo que no tiene interés para este proyecto ya que necesitamos que funcione en sistemas operativos sobre todo que no sean móviles, y mínimo en sistemas GNU/Linux.
- **HTK:** este software es de reconocimiento de voz, no realiza transcripciones que es lo que se necesitaría para el proyecto, quizá sería útil más en casos de IVR o similar, por lo que queda descartado.
- **ISIP:** al igual que HTK es para reconocimiento de voz y no transcripción. Además en este caso se trata de un proyecto obsoleto que no se ha continuado desarrollando y por ello mucha de la información acerca de éste ya no es accesible.
- **Julius:** también pensada para reconocimiento de voz, y entrenada en japonés lo que no nos serviría para VoIPCallMon. Puede trabajar en conjunto con VoxForge por ejemplo aportando la opción de transcripción, pero como se ha descrito anteriormente haría falta el entrenamiento de las mismas por lo que no es una opción buena en este caso.
- **Google Speech:** aunque funciona muy bien y los errores son mínimos, esta herramienta no realiza la transcripción de forma local sino online, por lo que queda descartada ya que por privacidad y seguridad la transcripción ha de realizarse localmente como se describe en los requisitos.
- **CMUSphinx:** este software cumple los requisitos deseados, pero se trata de un proyecto todavía en desarrollo, en fase pre-alpha según se puede ver al descargar el código de la herramienta en su página web [23], es decir, que aún puede tener muchos errores debido a que no se ha testeado y que todavía no está terminado. Por ello, no se trata de una herramienta fiable que podamos implementar en VoIPCallMon ya que sería algo que no es seguro que funcione.

Teniendo en cuenta los resultados de la evaluación de las librerías de transcripción de audio que hemos realizado, podemos concluir que, aunque algunas como Google Speech tienen una fiabilidad muy alta, ninguna se ajusta a todos los requisitos que se necesitan para VoIPCallMon. Por ello, se deja como trabajo futuro el desarrollo o la exploración de

otras herramientas que pudieran surgir y se adapten a los requisitos necesarios, comprobando la evolución de las que se han investigado en este TFG, para considerar las futuras mejoras de los desarrolladores y poder refinar así la comprobación de los requisitos funcionales y no funcionales necesarios para poder implementar el nuevo módulo de transcripción.

## *5.5 Conclusión*

Una vez realizadas las pruebas pertinentes, queda comprobado el correcto funcionamiento de las nuevas funciones de PDD y de buffer de tolerancia de pérdidas en VoIPCallMon. Por otro lado, quedan valoradas las librerías externas para transcripción de audio a texto de VoIPCallMon. Finalizado el trabajo a realizar en este proyecto, se exponen en el siguiente capítulo las conclusiones finales del mismo, lo aprendido a lo largo de dicho trabajo así como mejoras posibles para implementar en un futuro.

# 6 Conclusiones y trabajo futuro

## *6.1 Conclusiones*

Como resultados de este TFG, se han implementado, integrado y evaluado diversas mejoras en el sistema VoIPCallMon, que bien añaden nuevas funciones con el fin de que la herramienta sea lo más completa posible, bien mejoran los resultados de las funcionalidades que ya tenía el sistema. Para ello, se ha añadido la medición del tiempo PDD, un buffer de tolerancia a pérdidas de paquetes RTP, y por último se ha realizado una valoración de diversas librerías de estado del arte actual para transcripción de audio en texto.

Para presentar estos resultados, en esta memoria se han desarrollado una a una las mejoras a implementar, valorando los requisitos necesarios para cada una de ellas, indicando las principales decisiones de diseño y los detalles de la implementación, proporcionando una visión completa del desarrollo e integración de la mejora. Cada mejora, se ha implementado por separado siguiendo los requisitos descritos, de forma que posteriormente se ha realizado una comprobación de los resultados, y se ha mostrado y verificado que las mediciones obtenidas son las esperadas. Además, se ha comprobado (i) que las mejoras implementadas funcionan correctamente con diferentes escenarios que simulan las condiciones reales en una red de VoIP; (ii) que los resultados son los esperados y que efectivamente mejoran considerablemente el funcionamiento previo de la herramienta; y (iii) en el caso de la evaluación de las librerías de transcripción, se ha elaborado una comparativa y descripción de cada herramienta que permite verificar si cumple todos los requisitos de diseño del sistema que se quería extender. Con todo ello, se han elaborado también unas comparativas de herramientas de análisis y monitorización de tráfico de VoIP, y de las librerías de transcripción de audio, para mejorar la calidad de los sistemas que se apoyan en ellas.

Resumiendo, este trabajo ha conseguido cubrir todos los objetivos propuestos, introduciendo mejoras importantes en el sistema VoIPCallMon y siguiendo una metodología de evaluación que ha permitido comprobar cuantitativamente las aportaciones realizadas en este proyecto. Además, ha producido un conjunto de comparativas que facilitarán la evolución de este sistema en el futuro, y que pueden resultar de utilidad para otros profesionales de las TIC.

## *6.2 Trabajo futuro*

En este proyecto se pueden ampliar las funciones de la herramienta, y por ello se listan a continuación algunas sugerencias de mejora a implementar:

- Detectar en el buffer de tolerancia de pérdidas el códec que se está usando para que si no son los dos más usados, G.711 o G.729, se pueda ajustar el margen para considerar que un paquete se ha perdido.
- Valorar de nuevo las herramientas de estado del arte de las herramientas de transcripción y ver si se cumplen los requisitos deseados para el proyecto, con el fin de una futura posible implementación.
- Extender las funciones añadidas en este TFG para que puedan adaptarse a otros protocolos diferentes.

# Referencias


[1] J. L. García-Dorado, P.M. Santiago del Río, J. Ramos, D. Muelas, V. Moreno, J. E. López de Vergara, J. Aracil, "Low-cost and high-performance: VoIP monitoring and full-data retention at multi-Gb/s rates using commodity hardware". International Journal of Network Management, 24(3), 181-199, 2005

[2] "H.323 recommendation", ITU, https://www.itu.int/rec/T-REC-H.323/es

[3] J. F. Kurose, K. W. Ross, "Redes de Computadoras: Un enfoque descendente", Pearson, 2010.

[4] K. I. Park, "QoS in packet networks". Springer Science & Business Media. (2004).

[5] "PDD", Voip-info.org, 9 Julio 2012, http://www.voip-info.org/wiki/view/PDD

[6] J. A. Bergstra, C. A. Middelburg, "The e-model, a computational model for use in transmission planning". ITU-t recommendation G.107, 2003.

[7] R. G. Cole, J. H. Rosenbluth, "Voice over IP performance monitoring".ACM SIGCOMM Computer Communication Review, 31(2), 9-24, 2001.

[8] L. Deri, "Open Source VoIP traffic monitoring". Proceedings of SANE, Vol.2006.

[9] O. Hersent, J. P. Petit, D. Gurle, "Beyond VoIP Protocols: Understanding Voice Tecnology and Networking". 2005.

[10] "Netem", Ubuntu Manuals, http://manpages.ubuntu.com/manpages/wily/man8/tc-netem.8.html

[11] J. Ellis, C. Pursell, J. Rahman, "Voice, video, and data network convergence: architecture and design, from VoIP to Wireless". Academic Press, 2003.

[12] F. I. U. A. Khan, "A generic technique for voice over internet protocol (VoIP) traffic detection". IJCSNS, 8(2), 52, 2008.

[13] VoIPMonitor, http://www.voipmonitor.org/

[14] S. Karapantazis, F.N. Pavlidou, "VoIP: A comprehensive survey on a promising technology". Computer Networks, 53(12), 2050-2090, 2009.

[15] J. Saldana, J. Fernández Navajas, J. Ruiz-Mas, J. Murillo, E. V. Navarro, J. I. Aznar, "Evaluating the influence of multiplexing schemes and buffer implementation on perceived VoIP conversation quality". Computer Networks, 56(7), 1893-1919, 2012.

[16] H. Liu, P. Mouchtaris, "Voice over IP signaling: H.323 and beyond". IEEE Communications Magazine, 38(10), 142-148.

[17] V. N. Soares, P. A. Neves, J. J. Rodrigues, "Past, present and future of IP telephony". Communication Theory, Reliability and Quality of Service, CTRQ'08, Internacional Conference on, pp 19-24, IEEE, 2008.

[18] P. Montoro, E. Casilari, "A comparative study of VoIP Standards with Asterisk". Digital Telecommunications, ICDT'09, Fourth International Conference on, pp. 1-6, IEEE, 2009.

[19] F. F. Kuo, W. Effelsberg, J. J. Garcia-Luna-Aceves, "Multimedia communications protocols and applications". Prentice-Hall, Inc, 1998.

[20] M. Van der Schaar, P. A. Chou, "Multimedia over IP and Wireless networks: compression, networking, and systems". Academic Press, 2011

[21] A. Martín Hernández, "Desarrollo de un sistema de medida y recogida de datos y monitorización de tráfico VoIP". TFG, Julio-2014

[22] A. Nogales García, "Desarrollo de un sistema de medida y recogida de datos y monitorización de tráfico VoIP". TFG, Mayo-2015

[23] "CMUSphinx", Web de descarga, http://cmusphinx.sourceforge.net/wiki/download

# Glosario

| | |
|---|---|
| API | *Application Programming Interface* |
| HPCN | *High Performance Computing and Networking researching group* |
| IAX | *Inter Asterisk Exchange Protocol* |
| IP | *Internet Protocol* |
| IVR | *Interactive Voice Response* |
| MOS | *Mean Opinion Score* |
| PDD | *Post Dial Delay* |
| PSTN | *Public Switched Telephone Network* |
| RTP | *Real-Time Transport Protocol* |
| RTT | *Round-Trip delay Time* |
| SIP | *Session Initiation Protocol* |
| TIC | Tecnologías de la Información y la Comunicación |
| TCP | *Transmission Control Protocol* |
| UDP | *User Datagram Protocol* |
| VoIP | *Voice over IP* |

# Anexos

## A Scripts de pruebas para búffer de tolerancia a pérdidas

Para realizar las pruebas, se deben ejecutar (con permisos de superusuario) primero el script *Script_medidas.sh* y una vez acabado el proceso el script *Script_analisis.sh*. El primero generará las trazas, y ejecutará VoIPCallMon para obtener los resultados de las mediciones así como posteriormente VoIPMonitor. El segundo, extraerá los datos de los archivos generados por VoIPCallMon en un archivo incluyendo las medias y desviaciones típicas clasificadas por tiempo de retraso en red.
A continuación se muestran los scripts utilizados.

- Script_medidas.sh

```
#!/bin/bash

INTERFACE="lo"

EXEC="./RTPTracker"
MODE=1
SIGNALING_EXP=300
RTP_EXP=400
INPUT_PATH=TrazasPrueba
OUTPUT_PATH=Resultados
COLEC_RAW=0
COLEC_PCAPS=0
INPUT_FORMAT=pcap
CONFIG_FILE=params_RTPTracker.cfg

# Hay que ejecutar como root!

if [[ $1 == 'trazas' ]] ; then
# Delay adicional en ms

for d in 0 5 10 20 50 75 100 ; do

for ind in `seq 1 50` ; do

echo "Generando trazas con delay $d ms"

tc qdisc add dev $INTERFACE root netem delay "$d"ms "$d"ms 2> /dev/null #
Introduzco el retardo con el que quiero medir en la red

# Ponemos a capturar
tcpdump -i $INTERFACE -w RTP.pcap 2> /dev/null &

# Ponemos a reproducir
tcpreplay -i $INTERFACE RTPSR.pcap 2> /dev/null

sleep 1
# Cuando terminamos de reproducir, terminamos la captura
killall tcpdump

# Quito el retardo de red
tc qdisc del dev $INTERFACE root
```

```
# Ponemos a capturar otra vez
tcpdump -i $INTERFACE -w TrazasPrueba/"$d"_"$ind".pcap 2> /dev/null &

# Ponemos a reproducir otra vez
tcpreplay -i $INTERFACE SIP.pcap RTP.pcap BYE.pcap
sleep 1

# Cuando terminamos de reproducir, terminamos la captura
killall tcpdump

done;
done;
fi;
for d in 0 5 10 20 50 75 100 ; do

for ind in `seq 1 50` ; do
INPUT_FILE="$d"_"$ind".pcap
$EXEC -f $MODE -i $INPUT_FILE -s $SIGNALING_EXP -m $RTP_EXP -o
$OUTPUT_PATH -w $COLEC_RAW -p $INPUT_PATH -t $INPUT_FORMAT -c
$CONFIG_FILE -d $COLEC_PCAPS

voipmonitor -u root -r
~/Desktop/RTPTracker_install_package/TrazasPrueba/"$d"_"$ind".pcap -h
localhost -v 2 -p abcdefg #Analizamos las trazas con voipmonitor

done;
done;

# Obtenemos listado de capturas
ls TrazasPrueba/* > listado_traza_pruebas.ls
```

- Script_análisis.sh

```
#!/bin/bash


rm resultados_total.dat
rm estadisticos_jitter.csv

for d in 0 5 10 20 50 75 100 ; do



echo "Analizando trazas con delay $d ms"

#for ind in {1..2} ; do
for ind in `seq 1 50` ; do



awk -v delay=$d '{print delay","$32","$33}'
Resultados/records/"$d"_"$ind".pcap-RTPrecords.dat >>
resultados_total.dat


awk 'BEGIN{FS=","}{n[$1]=n[$1]+1; media1[$1]=media1[$1]+$2;
media2[$1]=media2[$1]+$3;  media21[$1]=media21[$1]+$2*$2;
media22[$1]=media22[$1]+$3*$3;}END{for (del in media1) {print
del","media1[del]/n[del]","(media21[del]-
media1[del]*media1[del]/n[del])/(n[del]-
1)","media2[del]/n[del]","(media22[del]-
media2[del]*media2[del]/n[del])/(n[del]-1)}}' resultados_total.dat >>
estadisticos_jitter.csv


done;
done;

echo "Fin"
```

Estos scripts están disponibles bajo demanda para poder realizar las pruebas pertinentes.